IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

Applicant(s): Gryaznov, et al	
Application No.: Unassigned	Group Art Unit: Unassigned
Filed: April 30, 2001	
Title: System and Method for Identifying a Macro Virus Family Using a Macro Virus Definitions Database	Examiner: Unassigned
Attorney Docket No.: 002.0160.01	

Assistant Commissioner for Patents
Washington, D.C. 20231

DRAWING TRANSMITTAL LETTER

Sir:

Enclosed herewith please find:

( )____ sheets of redlined drawing(s) which indicate proposed changes to the drawing(s). Upon approval of these proposed changes, formal drawing(s) will be submitted.

( )____ sheets of corrected formal drawing(s), as required by the Notice of Patent Drawings Objection (PTO-948) which accompanied the Office Action dated _____.

( )____ sheets of corrected formal drawing(s), as required by the Notice of Patent Drawing(s) Objection (PTO-948) and approved in the Notice of Allowability dated _____.

(X) 21 sheets of formal drawing(s), submitted with the enclosed Utility Patent Application.

Examiner's approval of the entry of these drawings is respectfully requested.

Respectfully Submitted,

By _____
Patrick J.S. Inouye, Esq.

Attorney/Agent for Applicant(s)
Reg. No. 40297

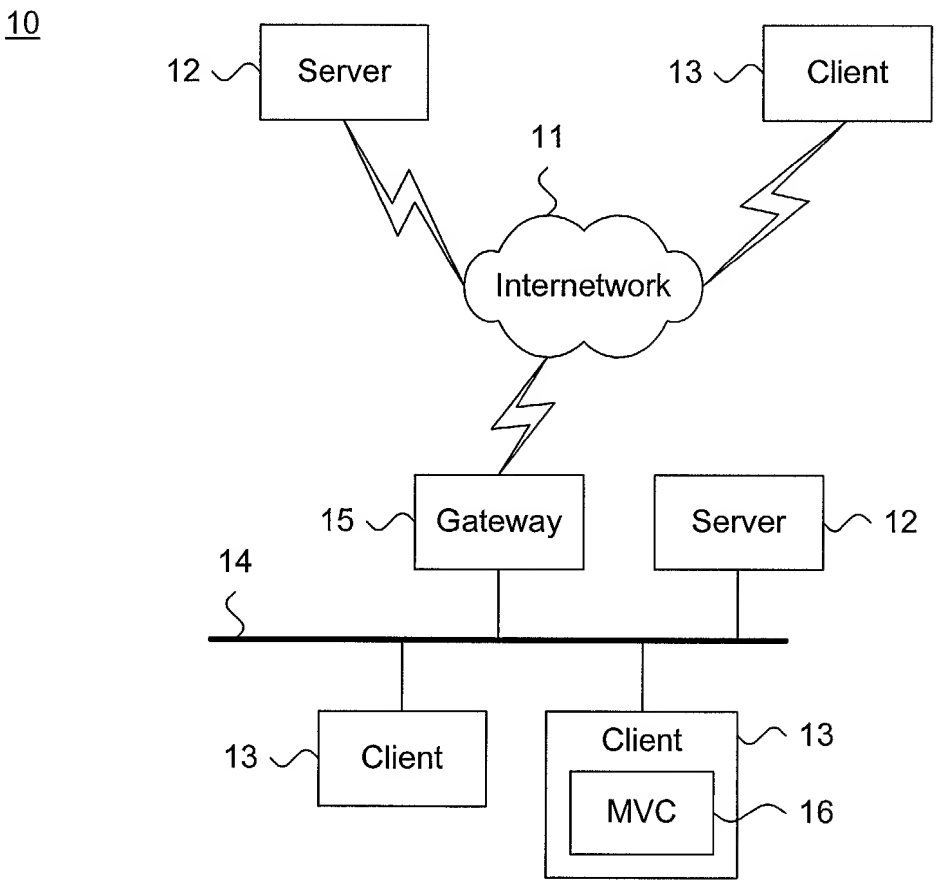
Date: April 30, 2001

Telephone No.: (206) 381-3900

DOCKET NO. 002.0160.01

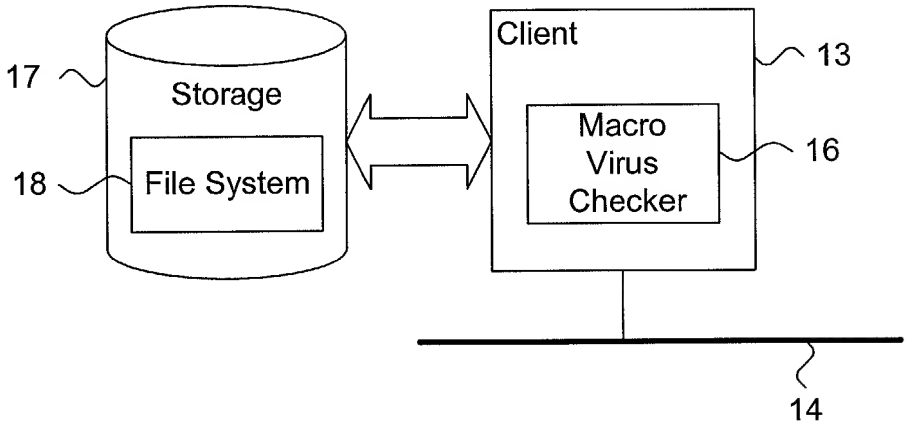
TODEtO" EOT9h860

Figure 1.



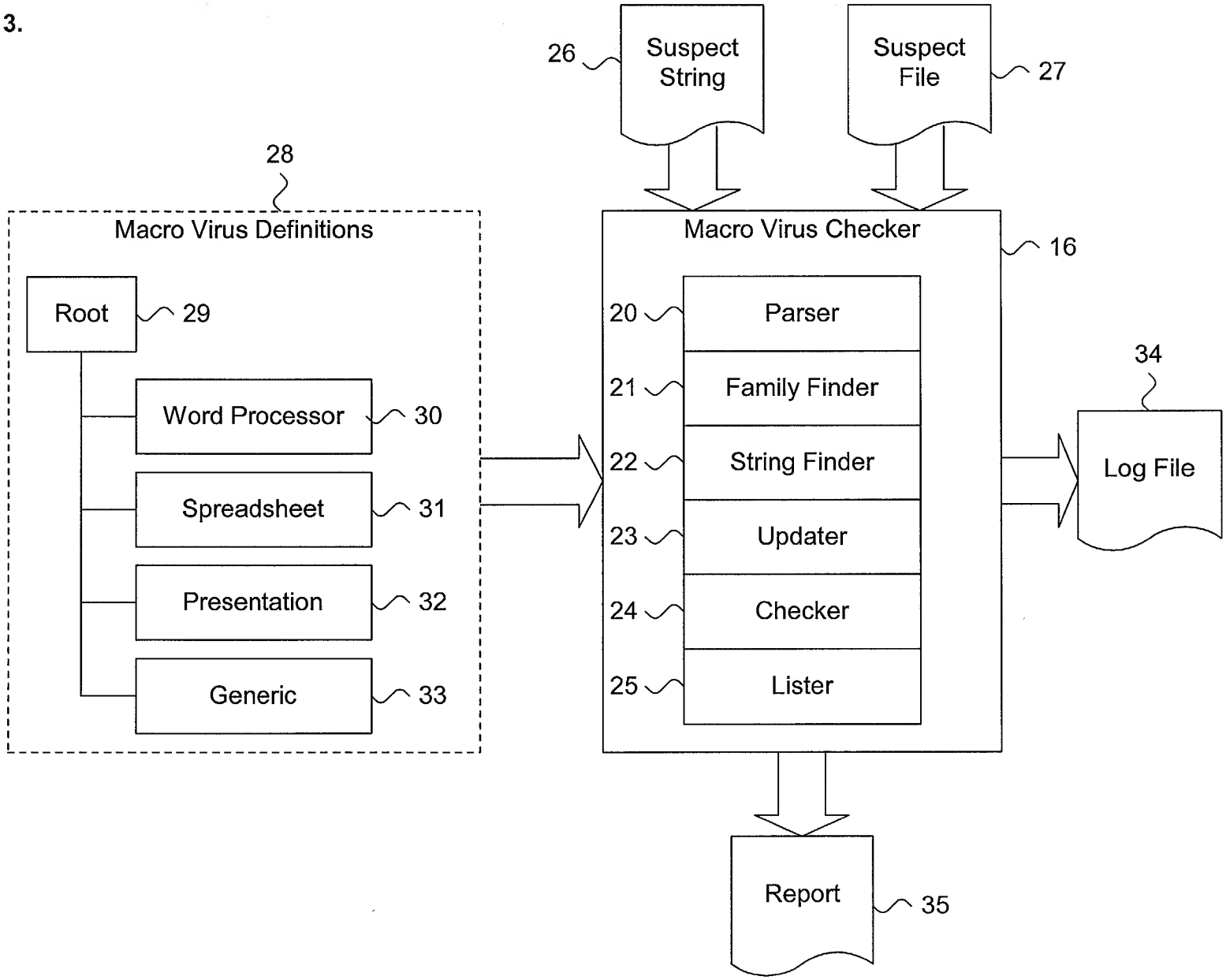
FOO2HO" EOT9H860

Figure 2.



TOOEhO" EOF9h860

Figure 3.



FOREIGN PATENT

Figure 4.

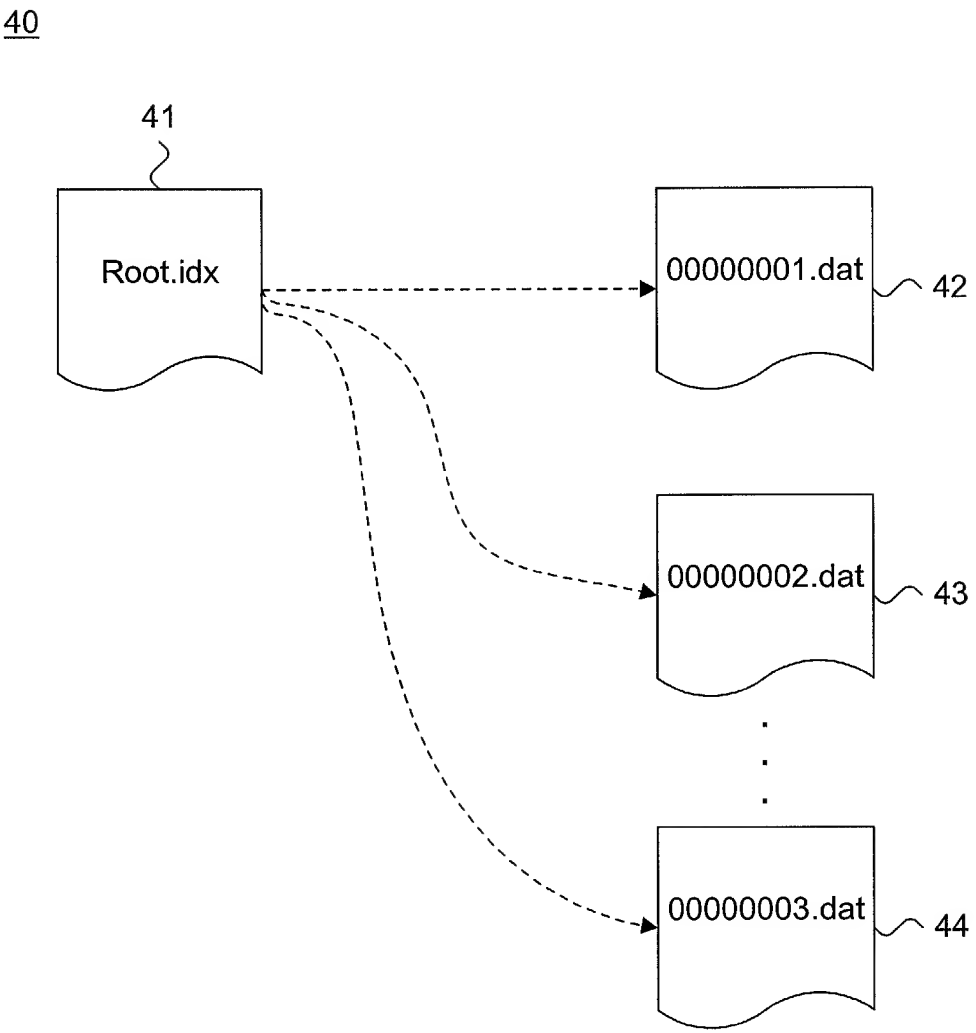


Figure 5.

50

```
typedef  struct  tagParseInfo{
        int                 FilesNum;
        TStrings            *Strings;
        TStrings            *Lines;
        int                 StringsNum;
        int                 TopString;          // index of first string in chain
        unsigned long       ReplFlags;
        int                 LinesNum;
        int                 TopLine;            // index of first index in chain
} TParseInfo;
```

Figure 6.

60

```
typedef  struct  tagStrings{
        char                *String;
        unsigned char       Type;
        unsigned char       Use;
        int                 Next;
} TStrings;
```

TOSHO" EOT94860

Figure 7.

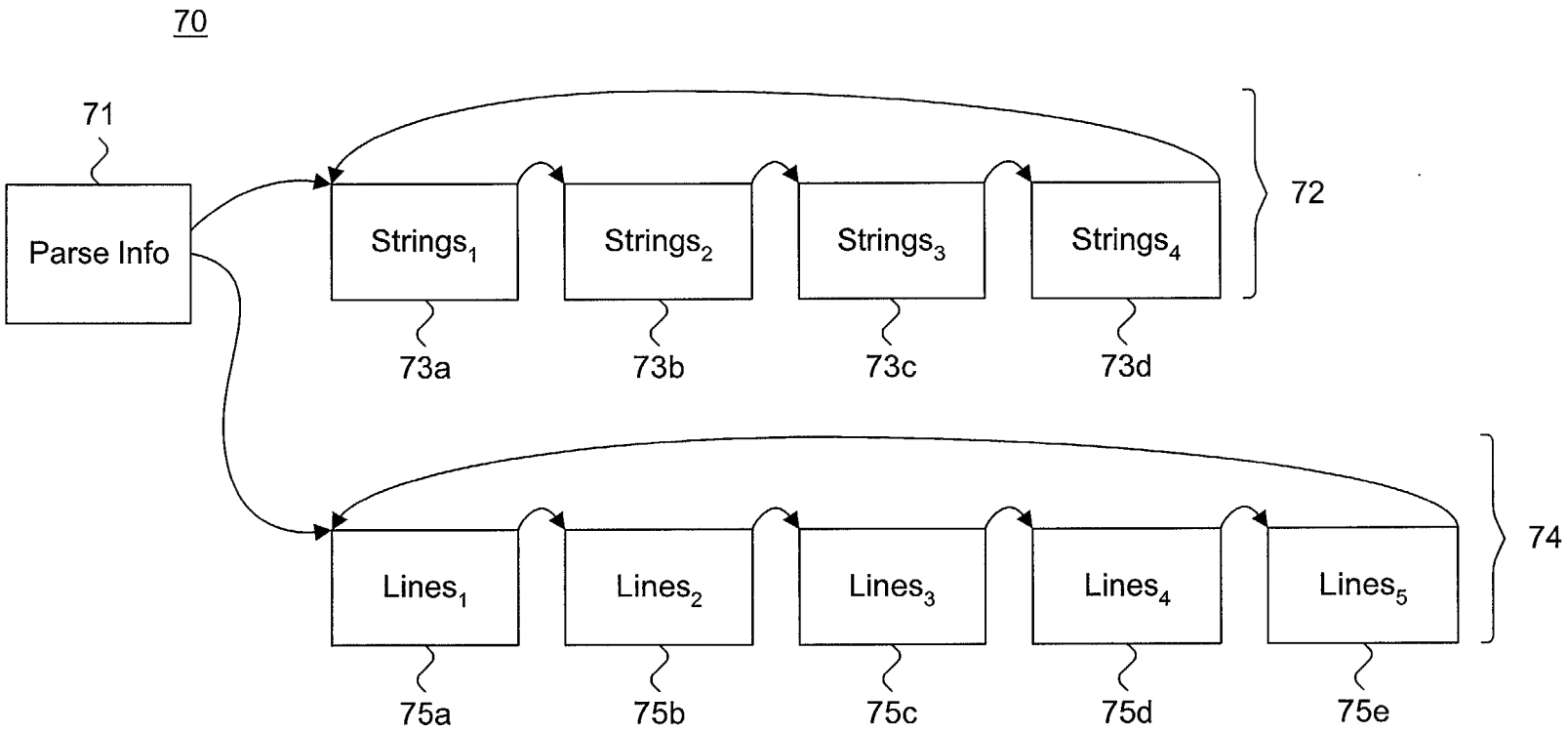


Figure 8.

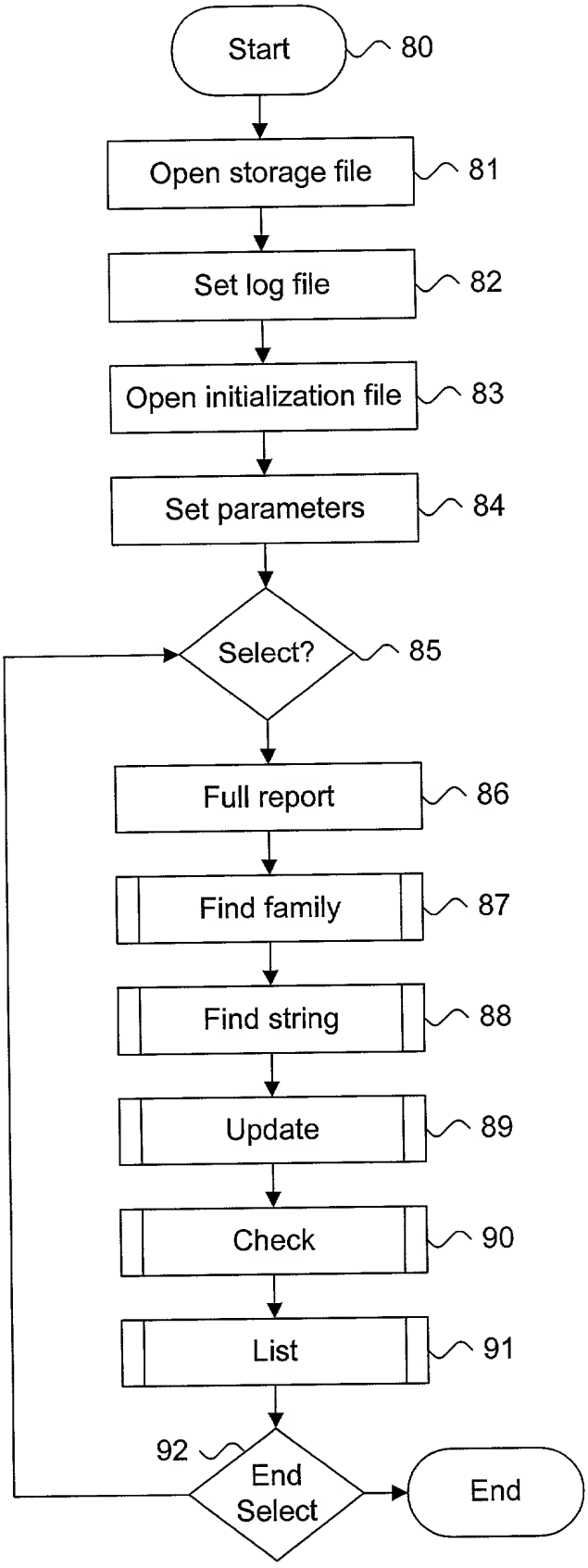

Start
80
Open storage file
81
Set log file
82
Open initialization file
83
Set parameters
84
Select?
85
Full report
86
Find family
87
Find string
88
Update
89
Check
90
List
91
92
End Select
End

Figure 9A.

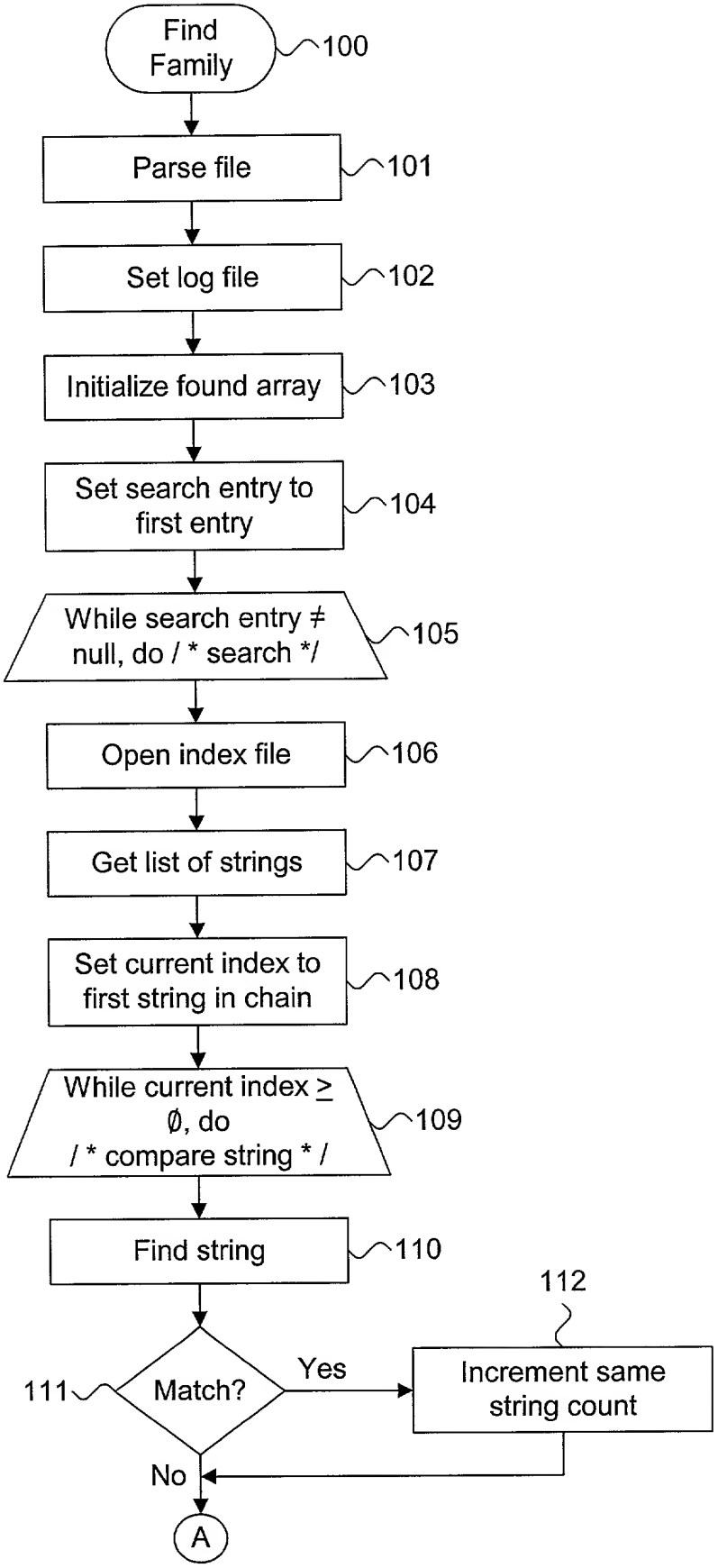

Find Family
100
Parse file
101
Set log file
102
Initialize found array
103
Set search entry to first entry
104
While search entry ≠ null, do / * search */
105
Open index file
106
Get list of strings
107
Set current index to first string in chain
108
While current index ≥ ∅, do / * compare string * /
109
Find string
110
112
111
Match?
Yes
Increment same string count
No
A

Figure 9B.

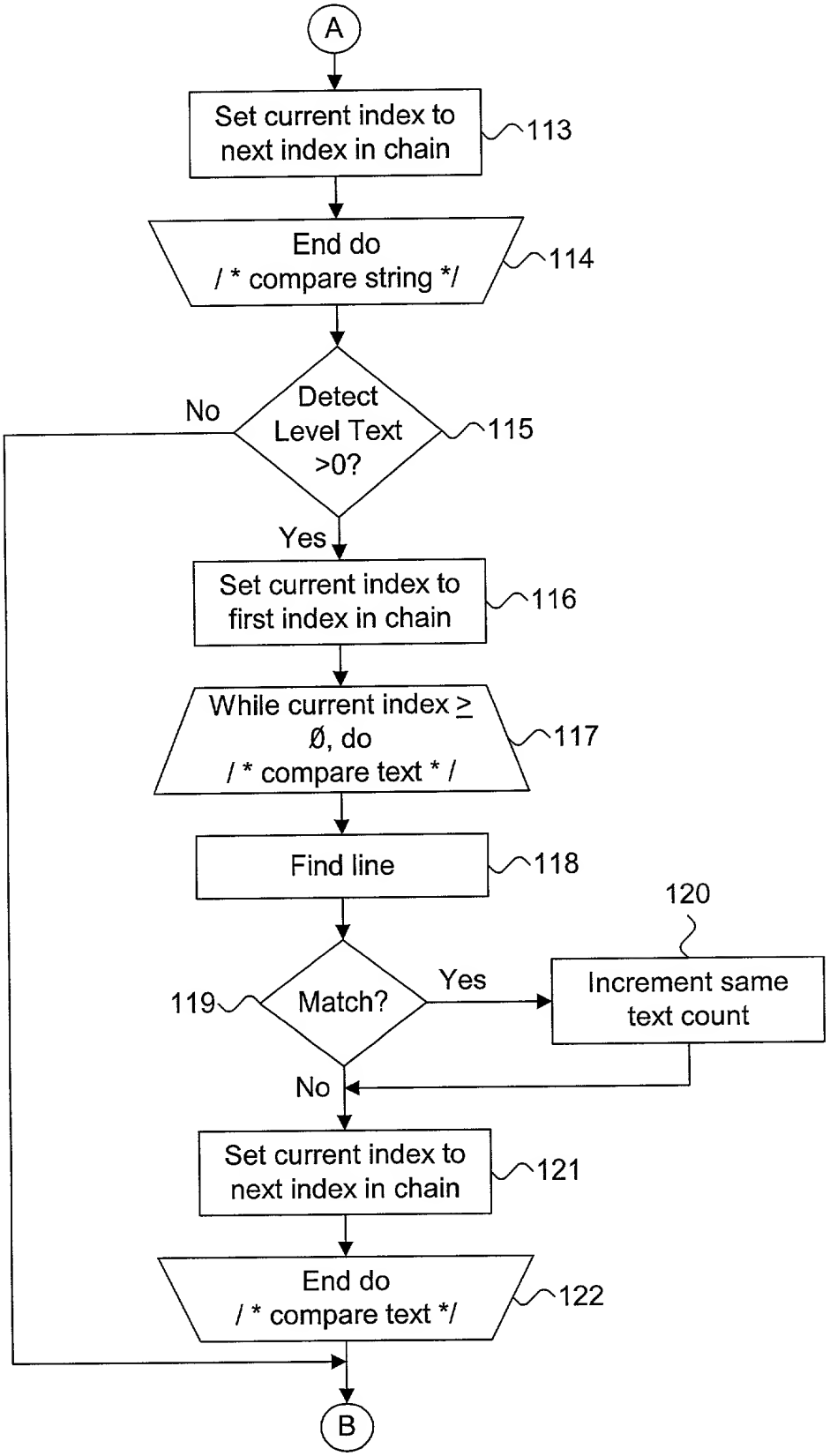

A
Set current index to next index in chain
113
End do / * compare string */
114
Detect Level Text >0?
115
No
Yes
Set current index to first index in chain
116
While current index ≥ Ø, do / * compare text * /
117
Find line
118
120
119
Match?
Yes
Increment same text count
No
Set current index to next index in chain
121
End do / * compare text */
122
B

Figure 9C.

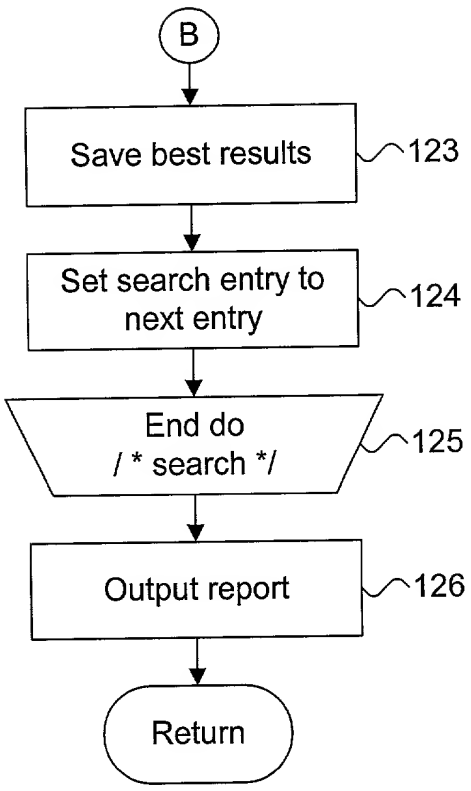

B
Save best results
123
Set search entry to next entry
124
End do / * search */
125
Output report
126
Return

Figure 10A.

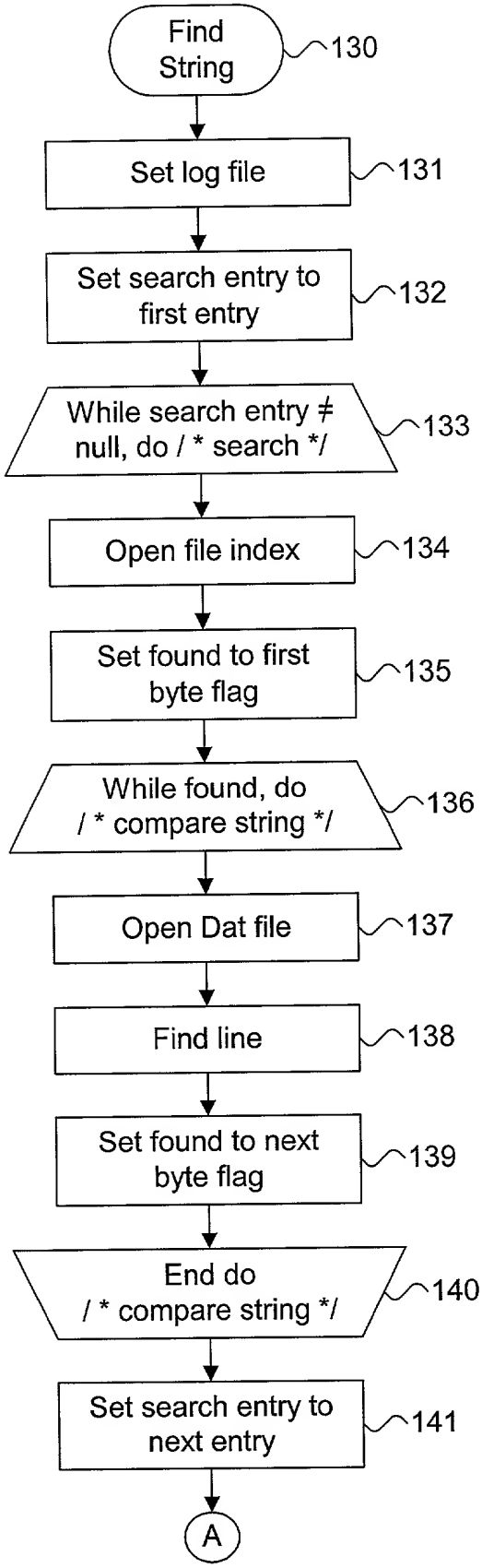

Find String
130
Set log file
131
Set search entry to first entry
132
While search entry ≠ null, do / * search */
133
Open file index
134
Set found to first byte flag
135
While found, do / * compare string */
136
Open Dat file
137
Find line
138
Set found to next byte flag
139
End do / * compare string */
140
Set search entry to next entry
141
A

Figure 10B.

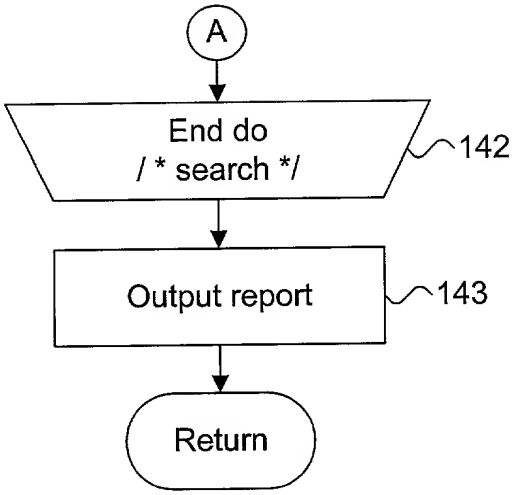


Figure 11A.

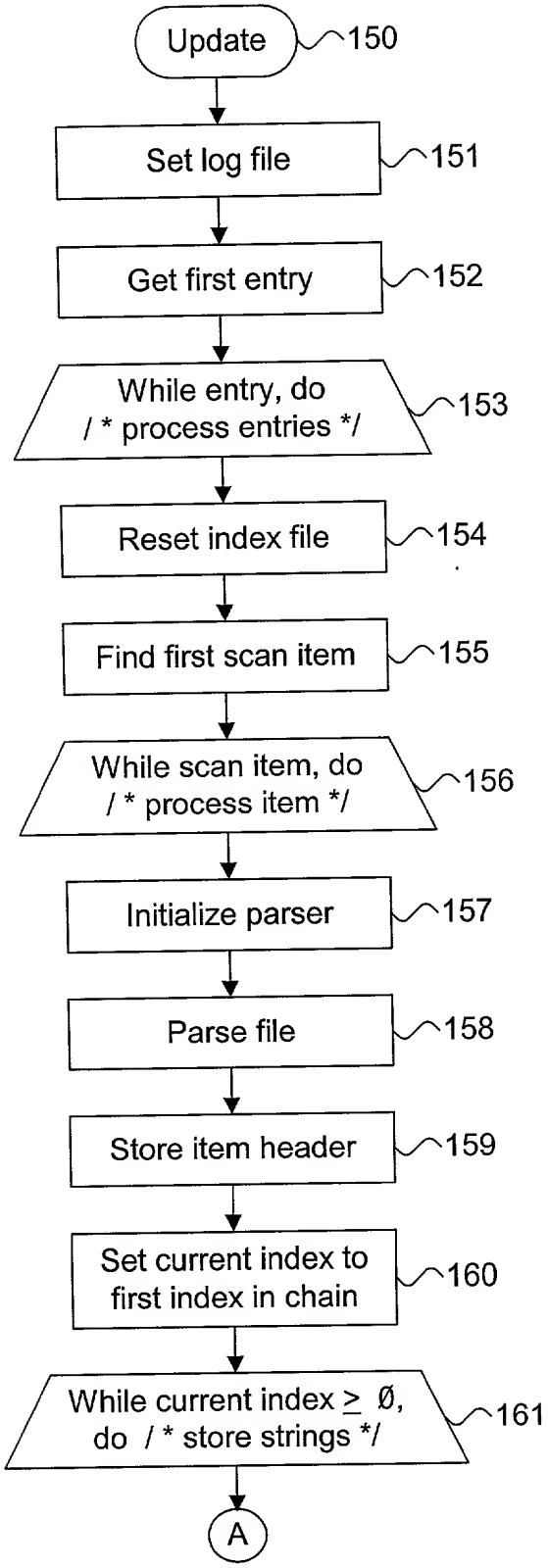

Update
150
Set log file
151
Get first entry
152
While entry, do
/ * process entries */
153
Reset index file
154
Find first scan item
155
While scan item, do
/ * process item */
156
Initialize parser
157
Parse file
158
Store item header
159
Set current index to
first index in chain
160
While current index ≥ Ø,
do / * store strings */
161
A

Figure 11B.

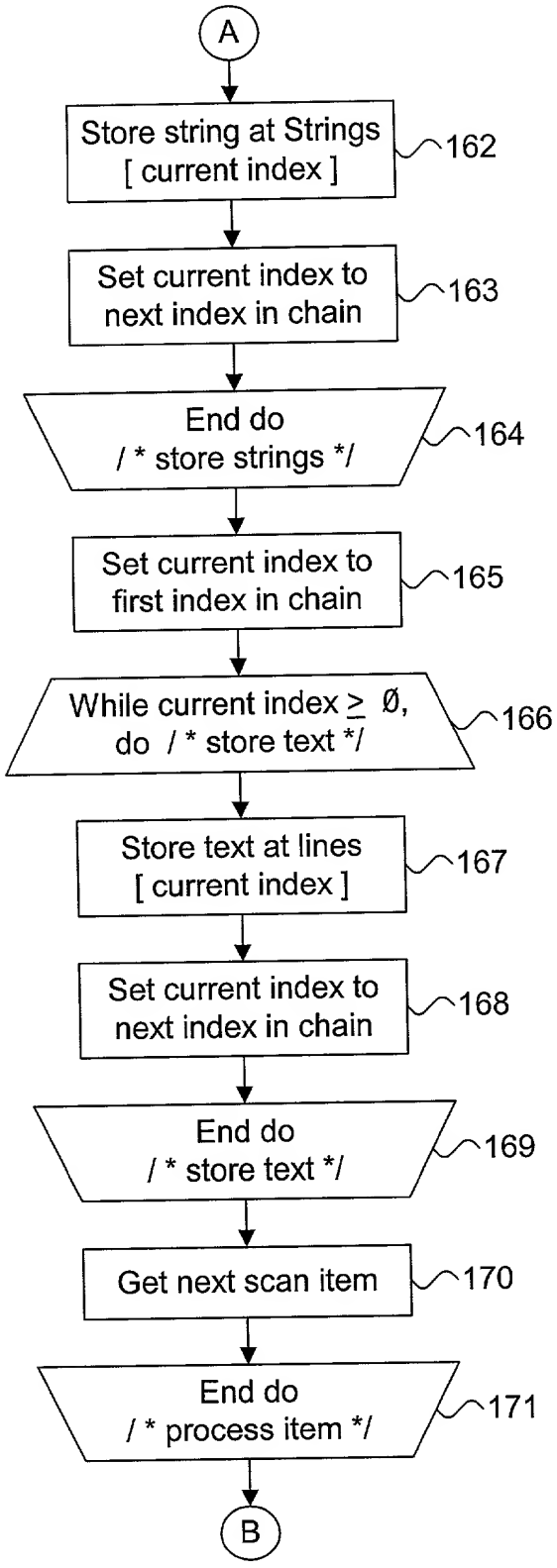

A
Store string at Strings [current index]
162
Set current index to next index in chain
163
End do / * store strings */
164
Set current index to first index in chain
165
While current index ≥ Ø, do / * store text */
166
Store text at lines [current index]
167
Set current index to next index in chain
168
End do / * store text */
169
Get next scan item
170
End do / * process item */
171
B

Figure 11C.

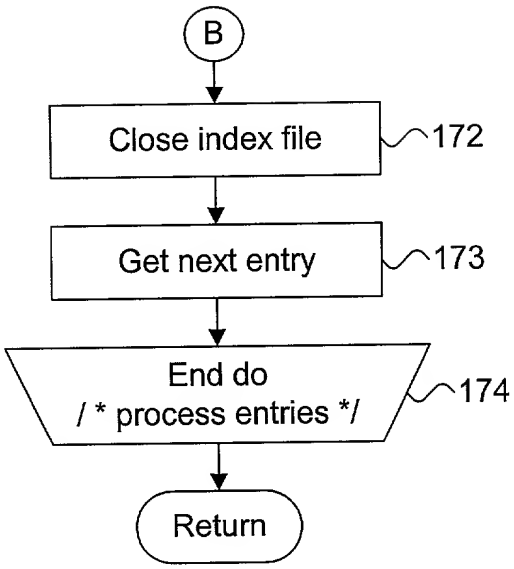

B
Close index file
172
Get next entry
173
End do
/ * process entries */
174
Return

Figure 12A.

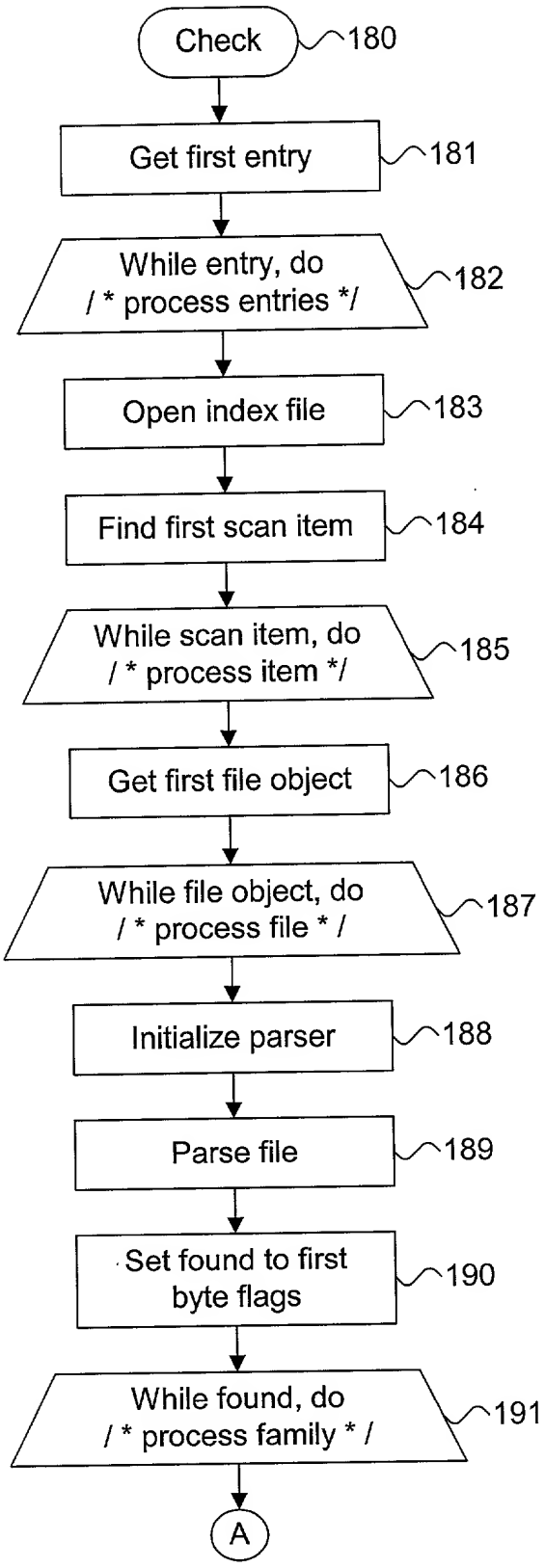

Check
180
Get first entry
181
While entry, do
/ * process entries */
182
Open index file
183
Find first scan item
184
While scan item, do
/ * process item */
185
Get first file object
186
While file object, do
/ * process file * /
187
Initialize parser
188
Parse file
189
Set found to first
byte flags
190
While found, do
/ * process family * /
191
A

Figure 12B.

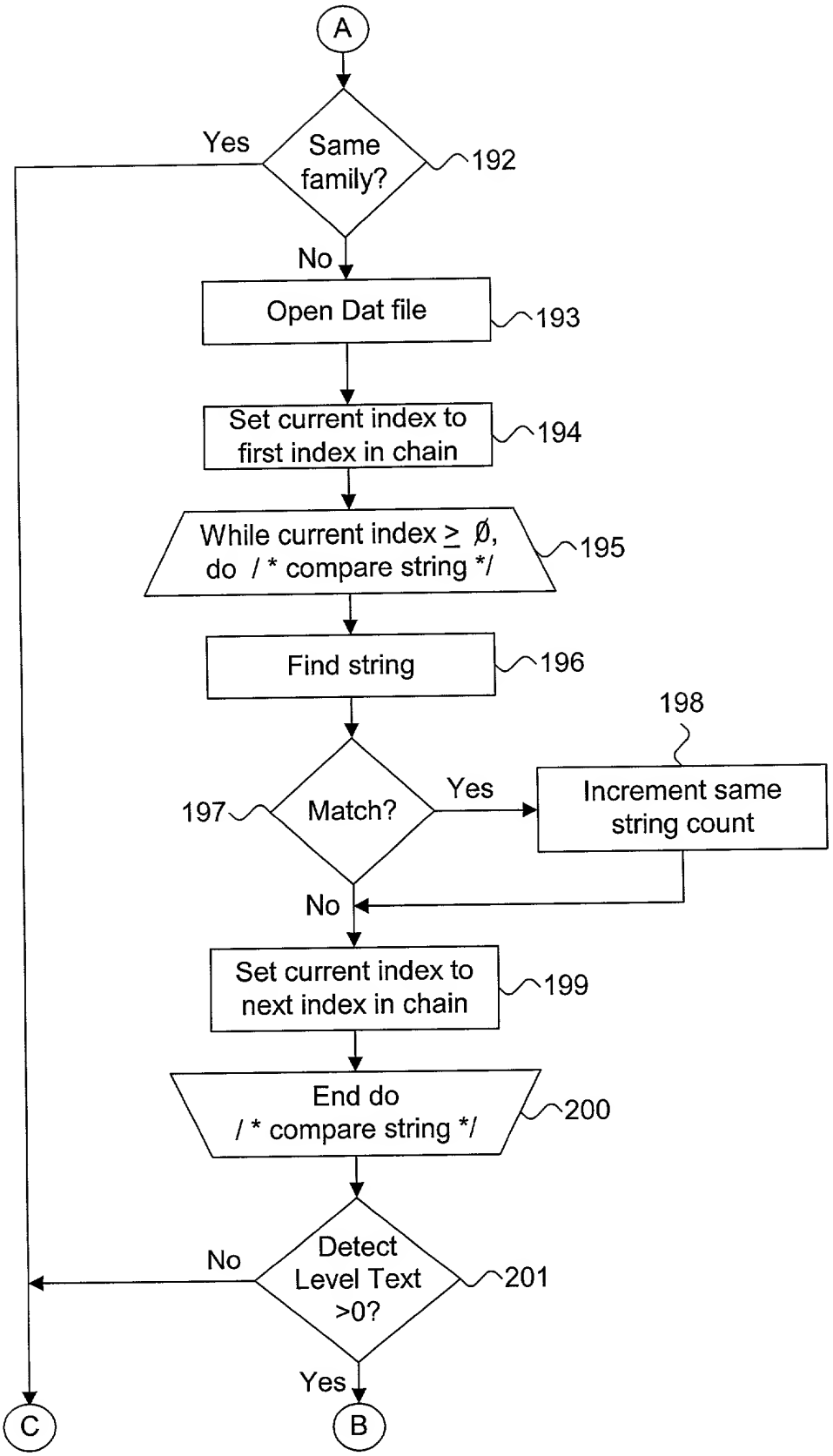

A
Same family?
Yes
192
No
Open Dat file
193
Set current index to first index in chain
194
While current index ≥ Ø, do / * compare string */
195
Find string
196
198
Match?
197
Yes
Increment same string count
No
Set current index to next index in chain
199
End do / * compare string */
200
Detect Level Text >0?
201
No
Yes
C
B

Figure 12C.

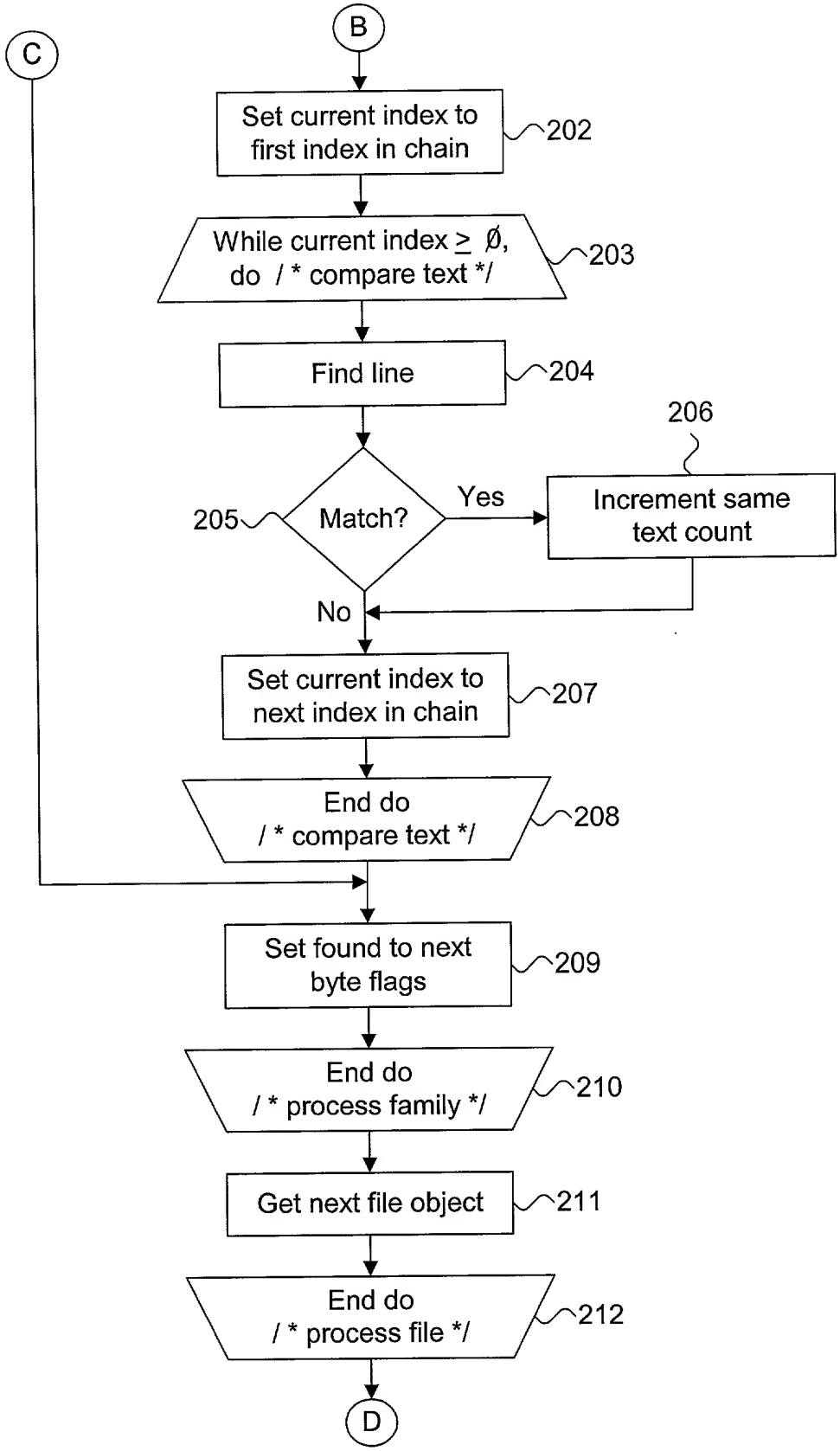
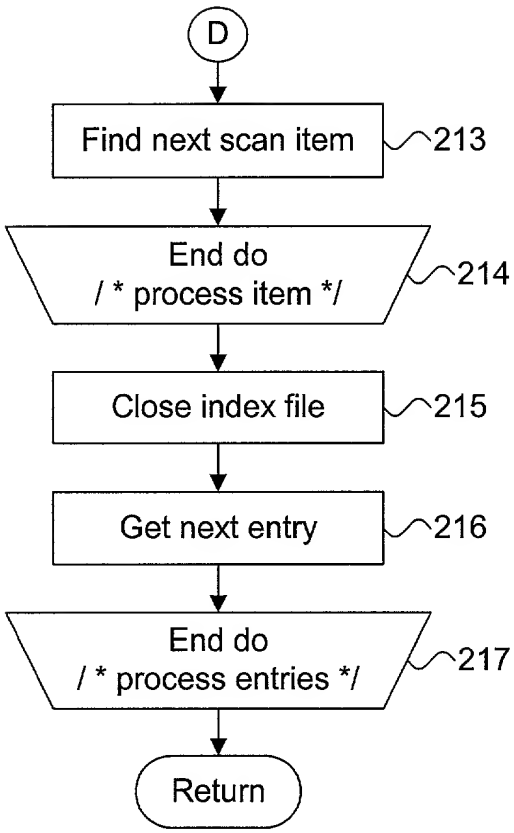

B
C
Set current index to first index in chain
202
While current index ≥ ∅, do /* compare text */
203
Find line
204
206
205
Match?
Yes
Increment same text count
No
Set current index to next index in chain
207
End do /* compare text */
208
Set found to next byte flags
209
End do /* process family */
210
Get next file object
211
End do /* process file */
212
D

Figure 12D.



Document 3 of 3 Page 3 of 3

Figure 13A.

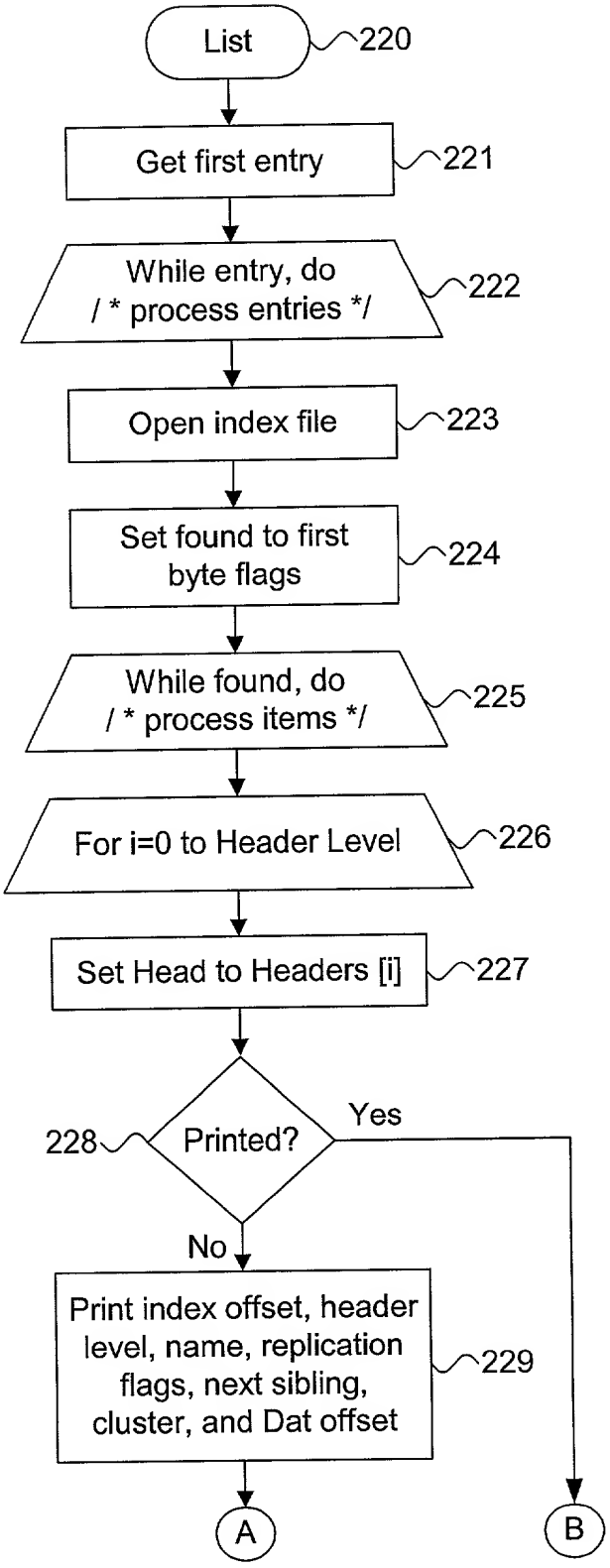
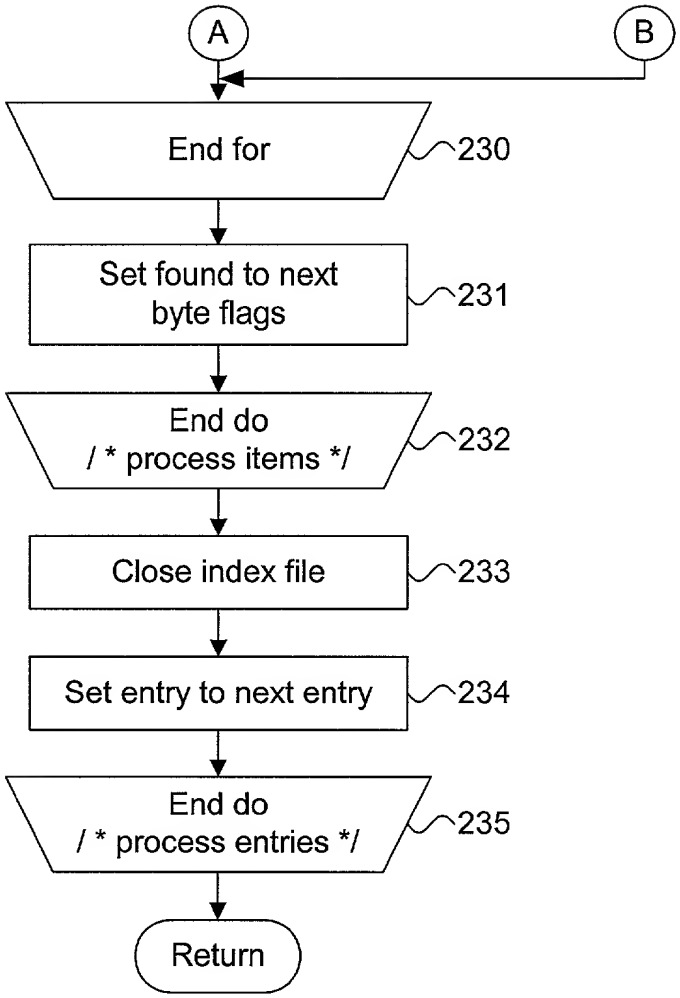

List
220
Get first entry
221
While entry, do
/ * process entries */
222
Open index file
223
Set found to first
byte flags
224
While found, do
/ * process items */
225
For i=0 to Header Level
226
Set Head to Headers [i]
227
228
Printed?
Yes
No
Print index offset, header
level, name, replication
flags, next sibling,
cluster, and Dat offset
229
A
B

Figure 13B.


A
B
End for
230
Set found to next byte flags
231
End do / * process items */
232
Close index file
233
Set entry to next entry
234
End do / * process entries */
235
Return